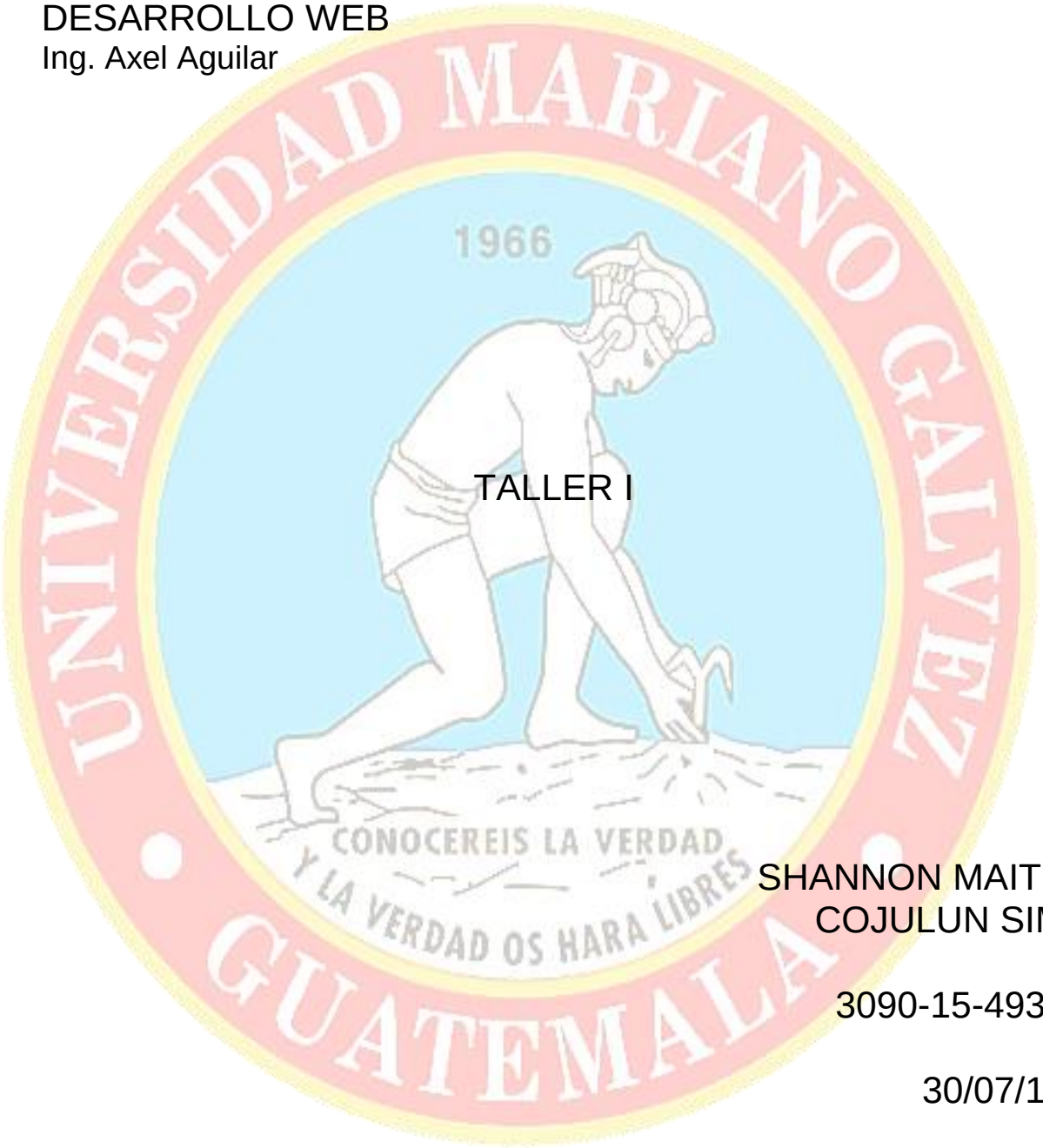
UNIVERSIDAD MARIANO GÁLVEZ

Mazatenango, Suchitepéquez.

INGENIERIA EN SISTEMAS DE INFORMACION
Y CIENCIAS DE LA COMPUTACION

DESARROLLO WEB

Ing. Axel Aguilar



TALLER I

SHANNON MAITE
COJULUN SIM

3090-15-4937

30/07/18

INTRODUCCION

Con las nuevas tecnologías que van surgiendo han creado la necesidad de conocer como funcionan los sistemas por capas, ya que van siendo vanguardia y muy utilizados en el campo de la informática.

El objetivo de este taller es crear un tutorial con los comandos más esenciales de cómo utilizar por consola un sistema por capas o versiones, dándole al lector una idea de cómo utilizar dicho sistema e introduciéndolo a nuevos conocimientos.

TALLER I
Servidores de Control de Versión y Versionador Git y GitHub

El objetivo del taller es explicar algunos comandos utilizados en Git para la utilización de los archivos en forma de capas.
Lo primero que debemos hacer es crear en el disco c una carpeta donde guardaremos los archivo a crear o que deseemos compartir.
Crearemos una carpeta llamada git y dentro de esta carpeta crearemos otra llamada master que es donde alojaremos los archivo a compartir.
Después de haber creado las carpetas y algunos archivos, nos vamos al cmd para poder entrar a git.

Primero como lo estamos utilizando por primera vez configuramos nuestro nombre y nuestro email. Utilizando el comando ***git config –globlal.***

```
C:\Users\Shannon>git config --global user.name "Shannon"

C:\Users\Shannon>git config --global user.email "<smcs-97@hotmail.com>"
```

Si deseamos ver la versión que tenemos de git, lo realizamos ingresando el siguiente comando. ***git versión***

```
C:\Users>git version
git version 2.18.0.windows.1
```

Luego ingresamos a la ubicación de las carpetas creadas anteriormente.

```
C:\Users>cd C:\git\master
```

El comando ***git init*** crea una carpeta en el repositorio en la ubicación que estamos, en la cual se almacenaran los archivo que deseemos compartir.

```
C:\git\master>git init
Initialized empty Git repository in C:/git/master/.git/
```

Con el comando ***git status*** podremos ver el estatus de nuestros archivos cuando salen en color rojo es debido a que no hemos agregado estos archivos a la carpeta y nos muestra que tenemos archivo que podemos agregar si deseamos.

```
C:\git\master>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        Taller.txt
        index.html

nothing added to commit but untracked files present (use "git add" to track)
```

Para agregar archivos a la carpeta utilizamos el comando ***git add -A***

```
C:\git\master>git add -A
```

Y si volvemos a dar un git status podremos ver como los archivos cambian de color.

```
C:\git\master>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   Taller.txt
        new file:   index.html
```

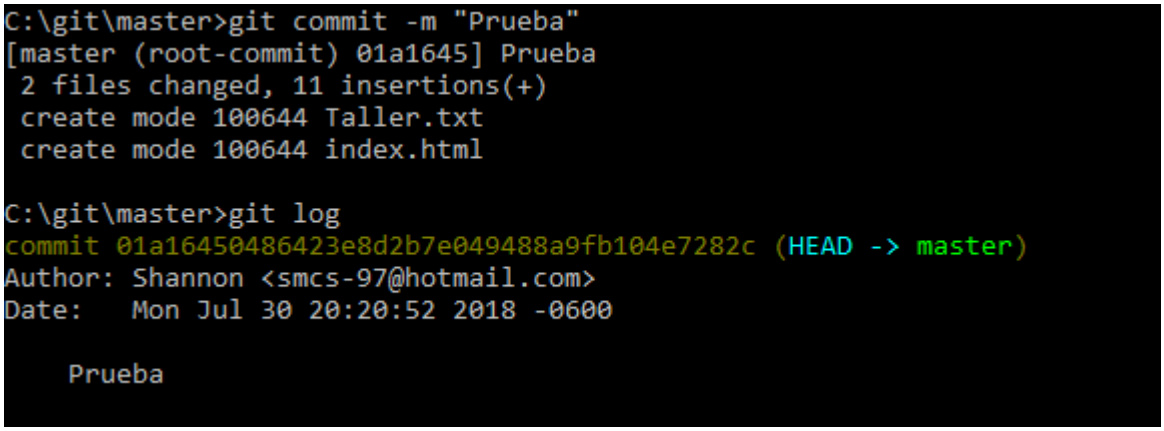
Otro comando que podemos utilizar es ***git add -help*** este comando nos permite entrar a la opción de ayuda de git donde nos mostrara diferentes opciones.

```
C:\git\master>git add -help
usage: git add [<options>] [--] <pathspec>...

    -n, --dry-run         dry run
    -v, --verbose         be verbose

    -i, --interactive     interactive picking
    -p, --patch           select hunks interactively
    -e, --edit            edit current diff and apply
    -f, --force           allow adding otherwise ignored files
    -u, --update          update tracked files
    --renormalize         renormalize EOL of tracked files (implies -u)
    -N, --intent-to-add   record only the fact that the path will be added later
    -A, --all             add changes from all tracked and untracked files
    --ignore-removal      ignore paths removed in the working tree (same as --no-all)
    --refresh             don't add, only refresh the index
    --ignore-errors       just skip files which cannot be added because of errors
    --ignore-missing      check if - even missing - files are ignored in dry run
    --chmod <(+/-)x>      override the executable bit of the listed files
```

Otros comandos que podemos utilizar son:

- ***git revert*** el cual sirve para regresar a una versión anterior
- ***git revert -help*** nos ayuda a buscar la opción para regresar al proceso que quiero.
- ***rm "nombredelarchivo"*** Elimina un archivo del sistema de archivos
  git rm "nombredelarchivo" Elimina un archivo del sistema y luego con un commit la acción se realizará.

Por último, tenemos el comando ***git commit -m ""*** con este comando guardamos cualquier cambio que hallamos realizado en la carpeta. git, ya sea que hallamos agregado, eliminado, modificado o restaurado algún archivo después de realizar la acción le damos un commit para guardar los cambios. Dentro de las comillas escribimos un mensaje donde podemos expresar que hemos hecho o solo para que sepan que hemos cambiado.



CONCLUSION

Los sistemas por capas o versiones están siendo muy utilizados en la actualidad debido a sus ventajas competitivas que da al momento de trabajar en equipo como al momento de estar actualizando archivos.

Se trato de explicar los comandos básicos del sistema GitHub el cual es un programa por capas que ayuda a trabajar en equipo, como a guardar las versiones y lograr a compartir los archivos de forma online.